CHAPTER 3

CHAPTER 3 *cdtime Module*

3.1 Time types

The cdtime module implements the CDMS time types, methods, and calendars. These are made available with the command

import cdtime

Two time types are available: *relative time* and *component time*. Relative time is time relative to a fixed base time. It consists of:

- a **units** string, of the form **'units since basetime**', and
- a floating−point **value**

For example, the time "28.0 days since 1996−1−1" has value**=28.0**, and units**='days since 1996−1−1'**

Component time consists of the integer fields **year, month, day, hour, minute**, and the floating−point field **second**. A sample component time is 1996−2−28 12:10:30.0

The **cdtime** module contains functions for converting between these forms, based on the common calendars used in climate simulation. Basic arithmetic and comparison operators are also available.

3.2 Calendars

A calendar specifies the number of days in each month, for a given year. cdtime supports these calendars:

- **cdtime.GregorianCalendar**: years evenly divisible by four are leap years, except century years not evenly divisible by 400. This is sometimes called the *proleptic* Gregorian calendar, meaning that the algorithm for leap years applies for all years.
- **cdtime.MixedCalendar**: mixed Julian/Gregorian calendar. Dates before 158210−15 are encoded with the Julian calendar, otherwise are encoded with the Gregorian calendar. The day immediately following 1582−10−4 is 1582−10−15. This is the default calendar.
- **cdtime.JulianCalendar**: years evenly divisible by four are leap years,
- **dtime.NoLeapCalendar**: all years have 365 days,
- **dtime.Calendar360**: all months have 30 days.

Several **cdtime** functions have an optional calendar argument. The default calendar is the **MixedCalendar**. The default calendar may be changed with the command:

cdtime.DefaultCalendar = *newCalenda*r

3.3 Time Constructors

The following table describes the methods for creating time types.

Table 3.1 Time Constructors

Type	Definition
Reltime	cdtime.reltime(value, relunits) Create a relative time type. value is an integer or floating point value. relunits is a string of the form "*unit(*s) [since *basetime*]" where unit = [second | minute | hour | day | week | month | season | year] basetime has the form yyyy−mm−dd hh:mi:ss. The default basetime is 1979−1−1, if no since clause is specified. Example: r = cdtime.reltime(28, "days since 1996−1−1")
Comptime	cdtime.comptime(year, month=1, day=1, hour=0,minute=0, second=0.0) Create a component time type. year is an integer. month is an integer in the range 1 .. 12 day is an integer in the range 1 .. 31 hour is an integer in the range 0 .. 23 minute is an integer in the range 0 .. 59 second is a floating point number in the range 0.0 ,, 60.0 Example: c = cdtime.comptime(1996, 2, 28)
Comptime	[Deprecated] cdtime.abstime(absvalue, absunits) Create a component time from an absolute time representation. absvalue is a floating−point encoding of an absolute time. absunits is the units template, a string of the form "unit as format", where unit is one of second, minute, hour, day, calendar_month, or calendar_year. format is a string of the form "%x[%x[...]][.%f]", where 'x' is one of the format letters 'Y' (year, including century), 'm' (two digit month, 01=January), 'd' (two−digit day within month), 'H' (hours since midnight), 'M' (minutes), or 'S' (seconds). The optional '.%f' denotes a floating−point fraction of the unit. Example: c = cdtime.abstime(19960228.0, "day as %Y%m%d.%f")

3.4 Relative Time

A relative time type has two members, **value** and **units**. Both can be set.

Table 3.2 Relative Time Members

Type	Name	Summary
Float	value	Number of units
String	units	Relative units, of the form **"unit(s) since basetime"**

3.5 Component Time

A component time type has six members, all of which are settable.

Table 3.3 Component Time Membersch3_cdms_4.0.html/#Table_3.1

Type	Name	Summary
Integer	year	Year value
Integer	month	Month, in the range 1..12
Integer	day	Day of month, in the range 1 .. 31
Integer	hour	Hour, in the range 0 .. 23
Integer	minute	Minute, in the range 0 .. 59
Float	second	Seconds, in the range 0.0 .. 60.0

3.6 Time Methods

The following methods apply both to relative and component times.

Table 3.4 Time Methods

Type	Definition
Comptime or Reltime	**t.add(value, intervalUnits, calendar=cdtime.Default−Calendar)** Add an interval of time to a time type t. Returns the same type of time. *value* is the Float number of interval units. *intervalUnits* is **cdtime.[Second(s) | Minute(s) | Hour(s) | Day(s) | Week(s) | Month(s) | Season(s) | Year(s)]** *calendar* is the calendar type. Example: **>>> from cdtime import *** **>>> c = comptime(1996,2,28)** **>>> r = reltime(28,"days since 1996−1−1")** **>>> print r.add(1,Day)** **29.00 days since 1996−1−1** **>>> print c.add(36,Hours)** **1996−2−29 12:0:0.0** Note: When adding or subtracting intervals of months or years, only the month and year of the result are significant. The reason is that intervals in months/years are not commensurate with intervals in days or fractional days. This leads to results that may be surprising. For example: **>>> c = comptime(1979,8,31)** **>>> c.add(1,Month)** **1979−9−1 0:0:0.0** In other words, the day component of **c** was ignored in the addition, and the day/hour/minute components of the results are just the defaults. If the interval is in years, the interval is converted internally to months:

	>>> c = comptime(1979,8,31) **>>> c.add(2,Years)** **1981−8−1 0:0:0.**0

Integer	t.cmp(t2, calendar=cdtime.DefaultCalendar)
	Compare time values t and t2. Returns −1, 0, 1 as t is less than, equal to, or greater than t2 respectively.
	
	t2 is the time to compare.
	calendar is the calendar type.
	Example:
	>>> from cdtime import *
	>>> r = cdtime.reltime(28,"days since 1996−1−1")
	>>> c = comptime(1996,2,28)
	>>> print r.cmp(c)
	−1
	>>> print c.cmp(r)
	1
	>>> print r.cmp(r)
	0

Comptime
or Reltime
	t.sub(value, intervalUnits, calendar=cdtime.DefaultCalendar)
	Subtract an interval of time from a time type t. Returns the same type of time.
	
	value is the Float number of interval units.
	intervalUnits is cdtime.[Second(s) | Minute(s) | Hour(s) | Day(s) | Week(s) | Month(s) | Season(s) | Year(s)]
	calendar is the calendar type.
	Example:
	>>> from cdtime import *
	>>> r = cdtime.reltime(28,"days since 1996−1−1")
	>>> c = comptime(1996,2,28)
	>>> print r.sub(10,Days)
	18.00 days since 1996−1−1
	>>> print c.sub(30,Days)
	1996−1−29 0:0:0.0
	For intervals of years or months, see the note under add().

Comptime	t.tocomp(calendar = cdtime.DefaultCalendar)
	Convert to component time. Returns the equivalent component time.
	calendar is the calendar type.
	Example:
	>>> r = cdtime.reltime(28,"days since 1996−1−1")
	>>> r.tocomp()
	1996−1−29 0:0:0.0

Reltime	t.torel(units, calendar=cdtime.DefaultCalendar)
	Convert to relative time. Returns the equivalent relative time.
	Example:
	>>> c = comptime(1996,2,28)
	>>> print c.torel("days since 1996−1−1")
	58.00 days since 1996−1−1
	>>> r = reltime(28,"days since 1996−1−1")
	>>> print r.torel("days since 1995")
	393.00 days since 1995
	>>> print r.torel("days since 1995").value
	393.0